Routing of Electric Vehicles: Constrained Shortest Path Problems with Resource Recovering Nodes*

Sören Merting[1], Christian Schwan[2], and Martin Strehler[2]

1 **Technische Universität München**
Boltzmannstr. 3, 85748 Garching near Munich, Germany
soeren.merting@in.tum.de

2 **Brandenburg University of Technology**
Platz der Deutschen Einheit 1, 03046 Cottbus, Germany
{christian.schwan, martin.strehler}@b-tu.de

Abstract

We consider a constrained shortest path problem with the possibility to refill the resource at certain nodes. This problem is motivated by routing electric vehicles with a comparatively short cruising range due to the limited battery capacity. Thus, for longer distances the battery has to be recharged on the way. Furthermore, electric vehicles can recuperate energy during downhill drive. We extend the common constrained shortest path problem to arbitrary costs on edges and we allow regaining resources at the cost of higher travel time. We show that this yields not shortest paths but shortest walks that may contain an arbitrary number of cycles. We study the structure of optimal solutions and develop approximation algorithms for finding short walks under mild assumptions on charging functions. We also address a corresponding network flow problem that generalizes these walks.

1998 ACM Subject Classification G.2.2 Graph Theory – Path and circuit problems, Network problems, G.2.3 Applications

Keywords and phrases routing of electric vehicles, constrained shortest paths, FPTAS, constrained network flow

Digital Object Identifier 10.4230/OASIcs.ATMOS.2015.29

1 Motivation

1.1 Electric Vehicles

Electric vehicles are a cornerstone towards eco-friendly mobility. Charged with renewable energy they contribute to a responsible use of our limited resources. Compared to common vehicles with combustion engines, there are still some disadvantages. The comparatively short range due to the restricted battery capacity is most likely the main reason for the poor popularization of electric vehicles up to now. Furthermore, for these cars even fast charging of the battery lasts significantly longer than traditional refueling.

This gives rise to many interesting mathematical questions concerning the routing of electric vehicles. First of all, electric vehicles can recuperate energy during downhill drive, i.e., there are road segments where energy consumption is negative. This also implies that for deciding whether a given path is feasible the remaining energy supply has to be checked

* This work was supported by the German Federal Ministry of Education and Research (BMBF), funding code 05M13ICA.

15th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS'15).
Editors: Giuseppe F. Italiano and Marie Schmidt; pp. 29–41

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

en route. Obviously, it must never be less than zero, but we also must not store more energy than the battery can hold in every point in time. Thus, is it possible to reach the destination? When no such path through the network can be found, battery charging stations have to be visited. But where should one charge the battery, especially when different charging stations provide different charging characteristics, e.g., rather slow charging at home compared to fast charging or even battery swapping at professional charging stations. How much energy should be regained at a certain charging station when charging costs (in time equivalents) are not proportional to the charged amount as the charge rate typically decreases at higher charge levels? All in all, which is the fastest feasible route to the destination?

There are also several fields of application for fleets of electric vehicles in commerce and industry. Exemplary, electric automated guided vehicles (agv) are operated in harbor terminals, manufacturing facilities or warehouses to move materials or containers around. Whereas reachability is of minor interest, charging all those vehicles may require a sophisticated planning. Thus, also the network flow version of the electric vehicle routing with limited charging capacity is worth studying.

1.2 Related Work

In the constrained shortest path problem (CSP), we are given a graph $G = (V, E)$ with a cost function $c : E \to \mathbb{R}$ and lengths or resources $r_i : E \to \mathbb{R}$. In general, costs and resources are assumed to be non-negative. Now, one seeks for a shortest path P from a vertex s to a vertex t which obeys the resource constraints R_i, $i \in N = \{1, \dots, k\}$. That is, we want to find an s-t-path P minimizing $\sum_{e \in P} c(e)$ such that $\sum_{e \in P} r_i \leq R_i \ \forall i \in N$.

For unit edge lengths, i.e., there is only one resource with $r(e) = 1$ for all edges, this problem can be solved easily using a labeling algorithm. In other words, we want to find a shortest path using at most R edges. The easiest way to solve this problem is a modified Bellman-Ford-algorithm which stops after R iterations. Here, it is important to update the labels simultaneously which automatically yields paths with no more than R edges.

The problem becomes $\mathcal{NP}$-complete when the resource function can take arbitrary non-negative values. This can be seen by a reduction of 2-Partition [10]. Thus, approaches focussing on listing all Pareto-optimal solutions via dynamic programming suffer from pseudopolynomial running times. Fortunately, several fully polynomial time approximation schemes (FPTAS) have been found to tackle this problem. Approaches based on *rounding and scaling* were suggested by Warburton [22] who used this technique to compute Pareto-optimal solutions for the multiple-objective shortest path problem. Hassin [13] adapted and improved this concept for constrained shortest paths. Shortly after, Phillips [18] presented an FPTAS which uses a Dijkstra search in a resource-expanded graph. In a second line of research, *multi-phase algorithms* were purposed. Most commonly, good lower and upper bounds are computed in a first phase and the remaining gap is closed in a second phase. Handler and Zang [12] used a Lagrange relaxation of an edge-based integer linear programming formulation to compute the lower and upper bounds. In the second phase, the gap is closed with help of a k-shortest path algorithm. Similarly, Beasley and Christofides [5] closed the gap with a branch and bound strategy. Another approach was proposed by Mehlhorn and Ziegelmann [15, 23]. They use the dual of a relaxed path-based integer linear programming formulation for the first phase. There, the separation problem can be solved efficiently which is used to compute the lower and upper bounds. For the second phase, additional approaches like path ranking and labeling strategies are discussed to close the duality gap.

A flow version of the constrained shortest path problem dates back to 1978, when Lovász et al. [14] derived a version of Menger's theorem for node-disjoint length bounded paths. This

was independently extended to edge-disjoint paths by Exoo [7] and Niepel and Šafaříková [17]. Edge-disjoint paths can be interpreted as a 0-1-valued flow with unit capacity and unit edge length. Recently, a more general approach with fractional flow values was studied by Baier et al. [1, 2]. Such a flow is feasible if and only if there exists a path decomposition such that each flow carrying path fulfills the resource constraint. However, the authors of [1, 2] show that it is $\mathcal{NP}$-complete to decide whether a given edge-flow can be decomposed into such paths. Consequently, deciding whether there is a length-bounded flow of a certain value is also an $\mathcal{NP}$-complete problem. The authors present a fully polynomial time approximation scheme based on an approximation algorithm for constrained shortest paths. Furthermore, the authors show that the ratio of the minimum fractional length-bounded s-t-cut, i.e., edges can be chosen partially, and the minimum integral s-t-cut can be of order $\Omega(\sqrt{n})$ even for unit resources. This bound carries over to the gap between fractional and integral flow.

The routing of electric vehicles has been studied to a lesser extent in the literature. In [19] the authors cope with negative resource consumption due to recuperation by taking the potential energy into account. Thus, a new non-negative cost function is determined such that an A* algorithm can be applied to compute paths with minimum energy consumption. Baum et al. [4] present a very fast routing algorithm for electric vehicles by extending the customizable route planning approach of [6]. Additionally, they consider reducing speed to increase range in a subsequent paper [3]. However, recharging at charging stations is not considered in these papers. Recently, a constrained shortest path problem for electric vehicles with recharging stations was introduced by Storandt [20]. In this setting, it is assumed that the whole battery is swapped at each charging point. Thus, recharge time is constant and the battery is always recharged to full capacity after each visit of a charging station. Consequently, first results were also obtained for the facility location problem for charging stations [21].

1.3 Our Contribution

In this paper, we extend previous results for resource constrained shortest paths and flows to networks with recharging nodes. Due to recuperation, negative resource consumption is now possible. Further, we consider recharging nodes where the resource can be refilled by paying additional costs.

The paper is organized as follows. Firstly, we give some definitions and fix the notation. Afterwards, we will show that in our setting shortest paths can contain cycles and even a node for charging can be visited more than once. Therefore, we also introduce a new type of conservative cost functions in order to avoid one class of cycles. In Section 4, we develop an FPTAS for the shortest path problem with charging. Furthermore, we address the flow variant of this problem, i.e., we present analytic results for the min cost flow problem with length constraints and recharging in Section 5.

2 Preliminaries

Throughout this paper the underlying structure is a *finite directed graph* $G = (V, E)$ with $n = |V|$ *vertices* or *nodes* and $m = |E|$ *edges*. Given a source vertex s and a target vertex t, an s-t-path P is a sequence of edges $(e_1 = (s, v_1), \ldots, e_k = (v_{k-1}, t))$ fitting head to tail and each edge appears only once. However, as we will show, paths are too restrictive for finding the most efficient route from s to t. An s-t-walk W is again a sequence of edges $(e_1 = (s, v_1), \ldots, e_k = (v_{k-1}, t))$, but each edge may appear more than once. A cycle C is a special kind of walk, where the first node is equal to the last one.

In general, one may allow of several resources, but here we limit our study to a cost function and a single resource constraint. For each edge $e \in E$, there are two parameters, namely *cost* $c : E \to \mathbb{R}_{\geq 0}$ and *resource consumption* $r : E \to \mathbb{R}$. Since costs are related to travel time in most of our applications, we assume the cost function to be non-negative. In contrast, we explicitly allow recuperation of energy. Thus, there may be edges with negative resource consumption. Of course, in accordance to basic laws of physics it is assumed that the total energy consumption on a cycle is non-negative.

▶ **Definition 1.** A resource function r is called *conservative* if for each cycle C there is

$$\sum_{e \in C} r(e) \geq 0.$$

In the following we require the resource function to be conservative. With our application in mind, we refer to a node as *charging node* if at this node the resource value of a path can be increased at the expense of the cost value . Furthermore, our main application requires the ability to model a non-linear charging process. We describe this *charging process* with help of a *charging function* $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ which is continuous and increasing and maps from the amount of recharged resources to the resulting costs.

▶ **Definition 2.** For a subset $S \subseteq V$ of *charging nodes*, the *charging function* $f_v : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, $v \in S$, is a continuous and increasing function. If the path P arrives at vertex $v \in S$ with x units of remaining resources we recharge μ resources which increase the cost of P by $f_v(x + \mu) - f_v(x)$.

This definition via the charging function has two advantages. Bypassing a charging node, that is $\mu = 0$, causes no additional cost. Further, there is also no need to interrupt the charging process, since $f_v(x+(\mu_1+\mu_2)) - f_v(x) = (f_v(x+\mu_1) - f_v(x)) + (f_v(x+\mu_1+\mu_2) - f_v(x+\mu_1))$. That is, we can assume that the desired amount of resources is recharged in one step. However, this does not prevent the optimal walk to visit a charging node $v \in S$ twice as we will show in Section 3.

▶ **Definition 3.** An *s-t-walk W with charging* is a sequence of tuples (e_i, μ_i), $i \in \{1, \ldots, k\}$ with edges $e_i = (v_{i-1}, v_i)$ and recharged resources $\mu_i \geq 0$ at v_i such that the edges form an s-t-walk. Furthermore, we require $v_i \in S$ if $\mu_i > 0$.

Contrary to constrained shortest paths, it is not sufficient to check the resource constraint only at the target terminal. Due to negative resource consumption on some edges, feasibility has to be checked at each intermediate point, too. Let us assume an initial resource value of R at node s and a lower bound of 0. Furthermore, let $\overline{R}$ be the maximum amount of storable resources. However, this upper bound is a soft bound. We will not violate it by recharging, since this will cause additional costs. If we would exceed this bound by recuperation, excess resources is not stored.

Let $r(W, j)$ be the remaining resources on walk W after the jth edge and after charging μ_j units of resources at node v_j. Starting with $r(W, 0) = R$ we define iteratively:

$$r(W, j) := \min\{\overline{R}, r(W, j-1) - r(e_j) + \mu_j\}.$$

▶ **Definition 4.** Given initial resources R and resource bound $\overline{R}$, a walk W with charging is feasible iff $r(W, j) - \mu_j \geq 0$ for all $j \in \{1, \ldots, k_W\}$.

Now, the cost of a walk W with charging is defined by

$$c(W) = \sum_{j=1}^{k} c(e_j) + \sum_{j=1,\ \mu_j > 0}^{k} f_{v_j}\left(r(W, j)\right) - f_{v_j}\left(r(W, j) - \mu_j\right).$$

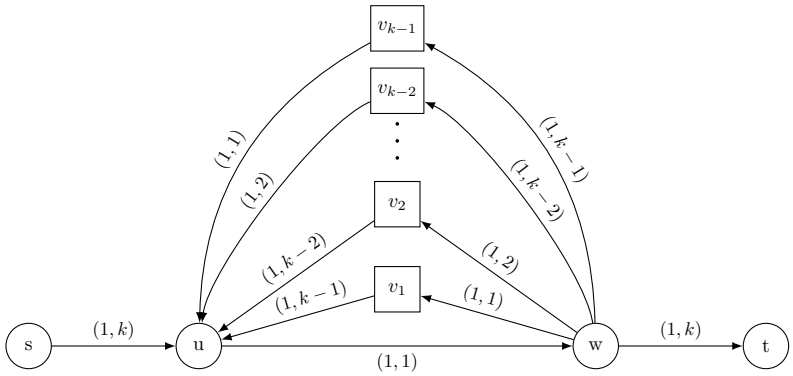

Figure 1 Network with common nodes (circle) and charging nodes (boxed). Edge labels denote *(cost, resource consumption)*. Initial resources and maximum storage are $R = \overline{R} = k + 2$. Charging is instantaneous/free, i.e., $f \equiv 0$ for all charging nodes, until reaching $\overline{R}$.

That is, the costs of a walk consist of costs for crossing the edges and of costs for recharging resources. Note that $r(W, j)$ already includes charging at the head of e_j, so resources are refilled from $r(W, j) - \mu_j$ up to $r(W, j)$. Let $r(W) = R - r(W, k_W)$ be the total resource difference of the walk W with k_W edges. A walk W_1 with charging is *dominating* a walk W_2 iff $c(W_1) < c(W_2)$ and $r(W_1) \leq r(W_2)$ or $c(W_1) \leq c(W_2)$ and $r(W_1) < r(W_2)$. Hence, a walk W_1 with charging is *Pareto-optimal* if there exists no walk W_2 which dominates W_1. A walk W_1 with charging is a *shortest walk*, if there is no other feasible walk W_2 with $c(W_2) < c(W_1)$.

3 Structure of Optimal Solutions

From an algorithmic point of view, the problem of deciding whether there is a walk with charging of cost $\leq c$ is clearly $\mathcal{NP}$-complete. On the one hand it is a decision problem and a walk given as certificate is easy to check, on the other hand the problem includes the constrained shortest path problem. Yet, in some sense it is also not harder than the constrained shortest path problem if we require an additional property of the network and its charging functions. We will derive this property in this section and we present a corresponding FPTAS in the next section.

However, the problem of finding shortest walks with charging has a richer combinatorial variety. In this section, we will also emphasize some of the most important properties of such walks. First of all, an optimal walk may contain cycles despite the positive cost function and the conservative resource consumption function.

▶ **Lemma 5.** *A shortest walk with charging may contain arbitrarily many cycles even when no charging costs exist. There are networks where an optimal walk passes a certain edge $\Omega(n)$-times.*

Proof. Consider the network in Figure 1. Charging node v_j, $j = 1, \ldots, k-1$ is reachable from charging nodes v_i with $i \geq j - 1$. Node v_j is only reachable from node v_{j-1} if the resources are completely recharged in v_{j-1}. Further, t is only reachable from v_{k-1}. Thus, even the shortest walk has to visit all charging nodes. Edge (u, w) is used k-times. ◀

The construction in Figure 1 may lead to the assumption that a shortest walk visits charging nodes at most once. However, this is not true.

▶ **Lemma 6.** *A shortest walk with charging can visit a single charging node several times.*

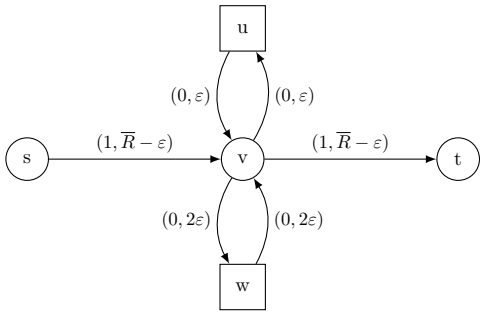

Figure 2 Network with two charging nodes. Charging is very expensive at u, that is $f_u(x) = x$. Charging at w is very fast/free, i.e., $f_w(x) = 0$. Even in this simple instance with linear charging functions, u is visited twice by the optimal path.

Proof. Again, we provide an example in Figure 2. Assume $R = \overline{R} = 1$. A walk starting in s may only reach v and u subsequently. From u, target t can only be reached by a full recharge $\mu_u = 1$. With $f_u(x) = x$, this also induces costs of 1. For small $\varepsilon > 0$, it is better to refill only 3ε. Then, w can be reached where completely charging is free. However, t is not reachable from w without visiting u and recharging 3ε again. Thus, if $\varepsilon < \frac{1}{6}$, the shortest path visits u twice. ◀

More than two visits of a charging node can easily be achieved by choosing the charging functions properly. Instead of a single free recharge node w as in Figure 2, assume several charging nodes w_i each providing free recharge only for a small subset of the domain of the charging functions and very high fees otherwise. Then, a shortest path can visit u several times. Actually, it is also possible to choose the charging function of two stations so that an optimal path would have to switch infinitely often between these two stations.

However, the network in the proof of Lemma 6 presumes somewhat unnatural charging functions. To prevent such unnatural walks, we introduce an additional constraint similar to the restrictions of conservative cost functions to prevent negative cost cycles.

▶ **Definition 7.** Let C be a cycle, let $S_C = S \cap V(C)$ be charging nodes on C, and let $v \in S_C$ be one of these nodes. Let W_C be the cyclic walk with charging, starting at node v with $r(W, 0) = R$ resources, using edges e_j, $j = 1, \ldots, k$ of C with arbitrarily recharged resources μ_j, and ending in node v. The cost of this walk is $c(W_C)$. Then, C is called a *regenerating cycle* iff there is a choice of recharged resources μ_j such that $r(W, k) > r(W, 0)$ and $c(W_C) < f_v(r(W, k)) - f_v(r(W, 0))$.

In other words, requiring a network without regenerating cycles implies that is always cheaper to recharge at a charging node immediately, instead of taking detours for recharging and coming back. Without charging, this definition would be equivalent to the definition of conservative resource functions. In accordance, we call the resource function and the charging functions *strictly conservative* if no regenerating cycles occur in the network.

Checking for such cycles is a difficult problem since there is an exponential number of cycles generally. Assuming non-linear charging functions, it is already $\mathcal{NP}$-hard to compute the optimal μ_i at each charging node on a single cycle since we may have to deal with a nonconvex nonlinear programming problem [16].

▶ **Theorem 8.** *In a network without regenerating cycles a shortest walk visits a charging node at most once.*

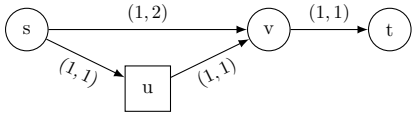

Figure 3 Assume $R = \overline{R} = 2$. Hence, an s-t-walk has to visit u for recharging. Obviously, the s-v-subwalk is not an optimal s-v-walk, since v can be reached directly from s with lower costs.

Proof. Definition 7 implies that visiting a charging node twice always causes additional costs (compared to charging at this node directly) or a loss of resources compared to the first visit. ◀

Please note that regenerating cycles are created by an interplay of resource functions and charging functions. Even in a network where all charging functions are identical, regenerating cycles may occur. Furthermore, identical charging functions do not imply either that it is optimal to completely charge the battery when a charging node is visited. Still, a path that visits several charging nodes and recharges small amounts of μ_i may be better than a path visiting fewer charging stations.

Finally, as shown next walks with charging do not have the shortest subwalk property.

▶ **Lemma 9.** *An s-v-subwalk of a shortest s-t-walk with charging is not necessarily a shortest s-v-walk.*

Proof. Figure 3 provides an instance without the subwalk property. Same claim also holds for constrained shortest paths and CSP is a subproblem of our problem. ◀

At first sight, the results in this section may seem of academic interest only. But charging functions for electric vehicles are non-linear. Consequently, such effects cannot be generally excluded in practice. Hence, each algorithm designed for this problem should be prepared to cope with cycles or one has to provide appropriate restrictions that are consistent with the corresponding application.

4 Approximating Constrained Shortest Walks with Charging

We will now develop an FPTAS for the constrained shortest walk problem with charging nodes. This will consist of two steps. Firstly, we construct an FPTAS for the common constrained shortest path problem but with conservative resource consumption and without charging at intermediate nodes. Secondly, we use a slightly modified version of the first algorithm as a sub-routine in an FPTAS connecting charging nodes in a network without regenerating cycles.

4.1 The Inner Approximation Algorithm

The inner approximation algorithm picks up the main idea of Hassin's approach, namely scaling and rounding [13], but we make some important changes to cope with negative resource consumption.

Let us fix two terminal nodes $\hat{s}$ and $\hat{t}$. Furthermore, we are given initial resources $\hat{R}$ and upper bound $\overline{R}$, a non-negative cost function $c : E \to \mathbb{R}_{\geq 0}$, and a conservative resource consumption $r : E \to \mathbb{R}$. In this inner approximation we do not consider recharging, i.e., there are no charging nodes. In consequence, no cycles can occur in an optimal path. Now, we want to find an $\hat{\varepsilon}$-approximation of the shortest s-t-path with respect to the cost function

c obeying the resource constraints. In other words, we want to find a feasible path that has cost at most $(1+\hat{\varepsilon})$-times higher than the optimal path.

To record information about subpaths, we provide a set of labels at each node. Each label consists of two values (c, r) of costs and remaining resources. Feasibility is crucial, hence, resource consumption has to be calculated exactly. Consequently, we can only round the cost values in the approximation algorithm. Here, the main difficulty is to find a suitable precision for rounding. On the one hand, the number of different values has to be bounded polynomially. On the other hand, we have to meet the approximation factor $1+\hat{\varepsilon}$.

Suppose that we would know the value OPT of the objective function of an optimal solution already. In this case, we can round up all cost values on the edges to integral multiples of $\frac{\hat{\varepsilon}\,\mathrm{OPT}}{|V|-1}$. Since the optimal path has at most $|V|-1$ edges, and the error on each edge is at most $\frac{\hat{\varepsilon}\,\mathrm{OPT}}{|V|-1}$, the total error of the path is at most $\hat{\varepsilon}\,\mathrm{OPT}$. If we round the cost values with this precision, we also get only integer multiples of $\frac{\hat{\varepsilon}\,\mathrm{OPT}}{|V|-1}$ as possible cost values for subpaths at each node. Since the length OPT of an optimal path is known and costs are non-negative, it suffices to record cost values up to this bound $(1+\hat{\varepsilon})\,\mathrm{OPT}$. Thus, we have at most $\frac{(|V|-1)(1+\hat{\varepsilon})}{\hat{\varepsilon}}$ different cost values that can occur at a node. Of course, for each node v we only store the set $Q(v)$ of Pareto-optimal labels, i.e., the highest battery charge achieved so far for each of the possible cost values.

To find a path of the desired length, we initially label node $\hat{s}$ with $Q(\hat{s}) = \{(0, \hat{R})\}$. All other label sets $Q(v)$ are initially empty. In an update step, we propagate the whole label set of each node u to all its neighbors v. Each label in $Q(u)$ is updated using cost and resource consumption of $e = (u, v)$ and added to the label set $Q(v)$ accordingly. Labels with cost values higher than $(1+\hat{\varepsilon})\,\mathrm{OPT}$ are discarded, they can never contribute to a path with the desired properties. Since an optimal path can consist of at most $|V|-1$ edges, precisely this number of update rounds suffices to find an optimal $\hat{s}$-$\hat{t}$-path according to Bellman-Ford's algorithm. Setting pointers for each label pointing to the predecessor node, the path itself can easily be reconstructed.

Thus, given a value OPT, we can check whether there is a feasible $\hat{s}$-$\hat{t}$-path of length at most $(1+\hat{\varepsilon})\,\mathrm{OPT}$ in time polynomial in $|V|$ and $\frac{1}{\hat{\varepsilon}}$. However, the OPT value is unknown at the beginning, so we apply binary search to find it. Let LB be the cost of a shortest path without considering the resource constraints. Thus, LB is a lower bound on OPT. Further, compute a shortest path with respect to resource consumption. Since resource consumption is conservative, one may use Bellman-Ford's algorithm. Obviously, the cost UB of this path is an upper bound on OPT. Our first guess on OPT is $\overline{\mathrm{OPT}} = \sqrt{\mathrm{LB}\cdot\mathrm{UB}}$, i.e., we use a logarithmic scale.

If we find a feasible path with length smaller than $\overline{\mathrm{OPT}}$, we can use this length as a new upper bound UB. If we cannot find a path of length at most $(1+\hat{\varepsilon})\overline{\mathrm{OPT}}$, we can use $\overline{\mathrm{OPT}}$ as a new lower bound LB. $\overline{\mathrm{OPT}}$ is updated accordingly and the binary search is stopped when $\frac{\mathrm{UB}}{\mathrm{LB}}$ is smaller than a pre-defined constant k, e.g., $k=2$. Now, we execute a final run with rounding precision $\frac{\hat{\varepsilon}\,\mathrm{LB}}{|V|-1}$ and maximum cost value UB. In other words, we use the precision given by the lower bound LB, but we use k-times as many labels to cover paths of length UB.

Summarizing, Algorithm 1 is a very short description of the inner approximation.

Algorithm 1 Inner Approximation.

Input: Graph $G = (V, E)$ with c, r and $\hat{\varepsilon}$, nodes $\hat{s}$ and $\hat{t}$, $Q(\hat{s}) = \{(0, \hat{R})\}$
Output: $\hat{s}$-$\hat{t}$-path with cost $\leq (1 + \hat{\varepsilon})\,\mathrm{OPT}$
compute LB, and UB, and $\overline{\mathrm{OPT}}$
while $\frac{UB}{LB} > 2$ **do**
    initialize Q, round c
    for $i = 1, \ldots, |V| - 1$ **do**
        for all nodes $v \in V$ **do**
            propagate $Q(v)$ to all neighbors
        end for
    end for
    update LB or UB, and $\overline{\mathrm{OPT}}$
end while
execute final run with double precision
if $Q(\hat{t}) \neq \emptyset$ **then**
    reconstruct $\hat{s}$-$\hat{t}$-path, return $Q(\hat{t})$
else
    no such path exists
end if

4.2 The Outer Approximation Algorithm

Whereas the inner approximation can be used for calculating costs and battery consumption between charging nodes, the outer approximation combines these paths between charging stations to create a walk from start node s to target node t.

The network for this algorithm consists only of the charging nodes S and any two nodes of S are connected if there is a feasible path between them in the original graph. However, reachability and the cost thereof depend on the initial battery charge. Thus, it is not possible to compute this reachability graph a priori. Instead, the cost for going from one charging node to another has to be computed just when needed.

Let $(1 + \varepsilon)$ be the accuracy of the outer approximation, we again determine the optimal rounding precision via a binary search on the optimal value $\mathrm{OPT}_{\mathrm{outer}}$ as described in Section 4.1. Only nodes $v \in S \cup \{s, t\}$ are labeled, initially all label sets are empty but $Q(s) = \{(0, R)\}$. To propagate a label from $u \in S \cup \{s\}$ to $w \in S \cup \{t\}$, we call the inner approximation with $\hat{s} = u$, $\hat{t} = w$, and $Q(\hat{s}) = Q(u)$.

In contrast to the pure inner approximation described above, we can omit the inner binary search by using the $\mathrm{OPT}_{\mathrm{outer}}$ value from the outer binary search and by setting $\hat{\varepsilon} = \frac{\varepsilon}{|S+1|}$. Thus, we also compute paths with higher costs but less consumption which match the required accuracy of the outer approximation. Note that this may lead to a poor actual approximation of the subpath. Especially if the subpath is very short, then an error of $\hat{\varepsilon}\,\mathrm{OPT}_{\mathrm{outer}}$ can be much larger than $\hat{\varepsilon}\,\mathrm{OPT}$. However, this relative error on subpaths does not matter as long as the total error of all subpaths is bounded.

Now, one has the option to charge the battery in node w. We use the return value $Q(\hat{t})$ of the inner approximation (where $\hat{t} = w$) and calculate all possible battery charges that match the cost discretization. Let f_w be the charging function at w, $(\hat{c}, \hat{r}) \in Q(\hat{t})$ and α the rounding precision of the outer approximation. We determine all values of x such that $0 \leq f(\hat{r} + x) - f(\hat{r}) + \hat{c} = k\alpha \leq (1 + \varepsilon)\,\mathrm{OPT}_{\mathrm{outer}}$ with $k \in \mathbb{N}$. Each label $(\hat{c}, \hat{r})$ of $Q(w)$ is shifted by all those values and is added to $Q(w)$ forming a new Pareto-optimal label set. Here, it is assumed implicitly that all operations concerning the computation of f can be done in polynomial time.

Since the resource function r is assumed to be strictly conservative, each charging node is visited at most once. We may also apply Bellman-Ford's principle here. Propagating the label set of each $u \in S \cup \{s\}$ to any other $w \in S \cup \{t\}$ in $|S| + 1$ rounds creates a

Algorithm 2 Outer Approximation.

Input: Graph $G = (V, E)$ with $S \subseteq V$, c, r and ε, nodes s and t, R
Output: s-t-path not longer than $(1 + \varepsilon)\,\mathrm{OPT}$
compute LB, and UB, and $\overline{\mathrm{OPT}}$
while $\frac{UB}{LB} > 2$ **do**
    initialize Q, $Q(s) = \{(0, R)\}$, round c
    for $i = 1, \ldots, |S| - 1$ **do**
        for all nodes $u \in V \cup \{s\}$ **do**
            for all nodes $w \in V \cup \{t\}$ **do**
                determine Pareto-optimal u-w-paths with maximum cost $\mathrm{OPT}_{\mathrm{outer}}$ and precision $\hat{\varepsilon} = \frac{\varepsilon}{|S+1|}$, and initial label set $Q(u)$
                apply recharging in w
                update $Q(w)$
            end for
        end for
    end for
    update LB or UB, and $\overline{\mathrm{OPT}}$
end while
if $Q(t) \neq \emptyset$ **then**
    reconstruct s-t-path, return $Q(\hat{t})$
else
    no such path exists
end if

correct label at t. Since we use at most $|S| + 1$ subpath and each subpath has at most error $\hat{\varepsilon}\,\mathrm{OPT} = \frac{\varepsilon}{|S+1|}\,\mathrm{OPT}$, the total error is less than $\varepsilon\,\mathrm{OPT}$.

Thus, in a very condensed form Algorithm 2 states the outer approximation scheme.

The outer approximation runs in time polynomial in $|S|$ and ε and the number of function calls to the inner approximation is also bounded polynomially. Due to the finer precision $\hat{\varepsilon} = \frac{\varepsilon}{|S+1|}$, the running time of the inner algorithm is increased by at most $\mathcal{O}(n)$ but the inner binary search is not needed. Please note that a feasible path for determining UB can be easily found by applying a full recharge at each charging node and checking reachability only for maximal resources. Due to space constraints, we have to omit a more detailed analysis but refer the reader to the journal version of this paper.

5 Flows with Recharging

Finally, we study the network flow version of constrained shortest paths in a network with charging nodes. In this setting, we are looking for a network flow that has a path decomposition such that each path is feasible with respect to the resource consumption. Hence, the constrained shortest path problem can be seen as a sub-problem. For example, the flow is zero iff no feasible path exists. Consequently, the flow problem is at least as difficult as the path problem.

We look at the unweighted version of this problem, here. That means, we only have capacities $u : E \to \mathbb{R}_{\geq 0}$ and resource consumption but no costs on the edges. The capacities limit the maximal flow on each edge. We also change the interpretation of charging nodes. A charging node now also has a capacity $c : S \to \mathbb{R}_{\geq 0} \cup \{+\infty\}$ which states the maximum amount of resources that can be provided for charging. One may also use a different kind of charging function to capture different efficiency factors during charging, i.e., the charging station has to expend more energy than arrives in the battery. In practice, this difference depends on the charge level of the battery. For simplicity, we assume a linear relation in this paper.

As seen in Section 3, cycles may occur. Since walks are the main actors, we use a walk based flow definition. Here, the parameter $\lambda_e(K)$ counts the number of occurrences of edge

e in a walk K. Further, $\mathcal{K}_{s,t}$ denotes the set of all possible s-t-walks. Note that due to the walk based formulation flow conservation is automatically implied.

▶ **Definition 10.** An *s-t-flow* in this setting is a function $f : \mathcal{K}_{s,t} \to \mathbb{R}_{\geq 0}$ assigning a *flow value* f_K to each s-t-walk K in G. The sum $\sum_{K \in \mathcal{K}_{s,t}} f_K$ is called the *s-t-flow value of f*. The flow f is *feasible* if it respects edge capacities, i.e.,

$$\sum_{K \in \mathcal{K}_{s,t} : e \in K} \lambda_e(K) f_K \leq u_e \qquad \forall e \in E.$$

Given initial resources R and resource bound $\overline{R}$, a resource constrained *s-t-flow with charging* is an s-t-flow, where every walk K with $f(K) > 0$ is a feasible s-t walk with charging with respect to R and $\overline{R}$.

An s-t-walk K with charging that charges μ units at node $v \in S$ and which is used by $f(K)$ flow units uses $f(K)\mu$ units of the capacity $c(v)$.

▶ **Definition 11.** An s-t-flow with charging is *feasible*, if the underlying s-t-flow is feasible and all walks with $f(K) > 0$ charge in total at most $c(v)$ units in node v for all $v \in S$.

▶ **Definition 12.** Given initial resources R and resource bound $\overline{R}$, an *s-t-cut with charging* is a set of edges $E' \subseteq E$ such that there is no feasible s-t-walk with charging in $G = (V, E \backslash E')$. The accumulated capacity of the edges in E' is called the *cut value* of E'.

Now, we consider maximum s-t-flows with charging, that is, flows where the flow value is maximum among all feasible s-t-flows with charging. Even in such maximum flows it can be necessary that some path visits a certain charging node more than once.

▶ **Corollary 13.** *There are instances of the maximum s-t-flows with charging problem, where a path contributing to a maximum s-t-flow with charging has to visit a charging node more than once.*

Proof. Consider the network in Figure 2. Let all edge capacities and $c(w)$ be infinite. Hence, the flow is only restricted by $c(u)$. If we visit node u only once, there is only one feasible path and at most $\frac{c(u)}{\overline{R}}$ units can be sent from s to t. If we recharge 3ε per flow unit, visit node w for a full recharge, and go back to u for another charge of 3ϵ, we can send $\frac{c(u)}{6\varepsilon}$ units from s to t. Thus, if we choose $\varepsilon < \frac{\overline{R}}{6}$, the claim follows. ◀

But even without limiting rechargeable resources, i.e., $c(v) = +\infty$ for all $v \in S$, flows with recharging significantly differ from common network flows. The network in Figure 1 implies two more corollaries for resource constrained flow with charging.

▶ **Corollary 14.** *The gap between the flow value and the cut value of resource constrained flow in a network with charging nodes and unlimited supply at charging nodes can be of order $\Omega(n)$, even for planar networks with unit capacities.*

Proof. Add unit capacities to the graph in Figure 1. Since there is only one feasible path, and this path uses one edge $\Omega(n)$-times, the claim follows. ◀

▶ **Corollary 15.** *The gap between resource constrained fractional flow and integral flow with charging in a network with unlimited supply in charging nodes is unbounded.*

Furthermore, there exist networks that do not allow for an integral flow greater than zero despite all capacities being integral. Nevertheless, the total fractional flow value can be integral and positive in those networks.

Proof. Figure 1 with unit capacities provides an instance where no positive integral flow is possible. Using several copies of this network in a parallel manner and scaling capacities would allow a fractional flow of value 1. ◀

6 Discussion

In this paper, we studied shortest paths and flows in a resource constrained setting where it is possible to refill resources at some nodes. We considered charging costs which depend on the charged amount. Thus, finding good paths is not only a question of reachability, but we have also to decide where and how much to charge. This additional combinatorial variety makes the problem significantly more difficult. Thinking of applications like routing of electric vehicles it seems very challenging to run an FPTAS on an onboard unit in admissible time.

This suggests further research into two directions. On the one hand, one may work on alternative approximation algorithms, e.g., based on routing algorithms in a condensed resource-expanded network (cf. [8]). On the other hand, the cycle constraints are crucial. Are there other ways to control regenerating cycles? Checking consumption and charging functions for being strictly conservative is difficult and expensive. Is it possible to perform this check using some kind of cycle basis instead of the whole set of cycles?

One may also extend the problem. For example, one may ask for the shortest path if no more than $(1 + \alpha)$ of the resources of the most resource efficient path can be spent. Further, not only the charging functions may depend on the initial value of the battery, but also the consumption itself may differ for full or nearly empty batteries. If this cannot be handled in a preprocessing step which parameterizes the battery linearly, this will lead to some kind of dynamic shortest path problem with recharging.

Another open question is the existence of an FPTAS for maximum flows with charging. Such an FPTAS can make use of the FPTAS for s-t-walks with charging. For example, one may try an approach like the algorithm of Garg and Könemann [11] with the extension of Fleischer [9] to approximative shortest paths. However, one will need consistent dual cost functions for the recharge capacities of the charging nodes. Thinking of fleets of vehicles, one may also extend the problem to include a time component. That is, charging stations only provide a limited number of slots for charging vehicles simultaneously. Therefore, recharging also requires scheduling.

References

1 Georg Baier. *Flows with Path Restrictions*. PhD thesis, TU Berlin, 2003.

2 Georg Baier, Thomas Erlebach, Alexander Hall, Ekkehard Köhler, Heiko Schilling, and Martin Skutella. Length-bounded cuts and flows. In *Automata, Languages and Programming*, LNCS 4051, pages 679–690. Springer Berlin Heidelberg, 2006.

3 Moritz Baum, Julian Dibbelt, Lorenz Hübschle-Schneider, Thomas Pajor, and Dorothea Wagner. Speed-consumption tradeoff for electric vehicle route planning. In *Proceedings of the 14th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'14)*, OpenAccess Series in Informatics (OASIcs), pages 138–151, 2014.

4 Moritz Baum, Julian Dibbelt, Thomas Pajor, and Dorothea Wagner. Energy-optimal routes for electric vehicles. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 54–63. ACM Press, 2013.

5 John E. Beasley and Nicos Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19(4):379–394, 1989.

6 Daniel Delling, Andrew V. Goldberg, Thomas Pajor, and Renato F. Werneck. Customizable route planning. In P. M. Pardalos and S. Rebennack, editors, *Proceedings of the 10th International Symposium on Experimental Algorithms (SEA'11)*, LNCS 6630, pages 376–387. Springer, 2011.

7 Geoffrey Exoo. On line disjoint paths of bounded length. *Discrete Mathematics*, 44(3):317–318, 1983.

8 Lisa Fleischer and Martin Skutella. Quickest flows over time. *SIAM Journal on Computing*, 36(6):1600–1630, 2007.

9 Lisa K. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM Journal on Discrete Mathematics*, 13(4):505–520, 2000.

10 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

11 Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing*, 37(2):630–652, 2007.

12 Gabriel Handler and Israel Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4):293–309, 1980.

13 Refael Hassin. Approximation schemes for the restricted shortest path problem. *Math. Oper. Res.*, 17(1):36–42, February 1992.

14 László Lovász, Víctor Neumann-Lara, and Michael Plummer. Mengerian theorems for paths of bounded length. *Periodica Mathematica Hungarica*, 9(4):269–276, 1978.

15 Kurt Mehlhorn and Mark Ziegelmann. Resource constrained shortest paths. In Mike S. Paterson, editor, *Algorithms – ESA 2000*, LNCS 1879, pages 326–337. Springer Berlin Heidelberg, 2000.

16 Katta G. Murty and Santosh N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39(2):117–129, 1987.

17 Ludovít Niepel and Daniela Šafaříková. On a generalization of Menger's theorem. *Acta Mathematica Universitatis Comenianae*, 42:275–284, 1983.

18 Cynthia A. Phillips. The network inhibition problem. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing*, STOC'93, pages 776–785, New York, NY, USA, 1993. ACM.

19 Martin Sachenbacher, Martin Leucker, Andreas Artmeier, and Julian Haselmayr. Efficient energy-optimal routing for electric vehicles. In *Conference on Artificial Intelligence, Special Track on Computational Sustainability*. AAAI, 2011.

20 Sabine Storandt. Quick and energy-efficient routes: computing constrained shortest paths for electric vehicles. In *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pages 20–25. ACM, 2012.

21 Sabine Storandt and Stefan Funke. Enabling e-mobility: Facility location for battery loading stations. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, 2013.

22 Arthur Warburton. Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1):70, 1987.

23 Mark Ziegelmann. *Constrained shortest paths and related problems*. Phd thesis, Universität des Saarlandes, Saarbrücken, 2001.